REMARKS

In view of the foregoing amendments and following remarks responsive to the Office Action of January 31, 2005, Applicant respectfully requests favorable reconsideration of this application.

Applicant respectfully thanks the Examiner for the indication that claims 8-19, 25, 26, and 31-57 are merely objected to, but would be allowable if amended into independent form to include the limitations of their base claims.

Only claims 1-7, 20-24, and 27-30 stand rejected. In particular, as in the previous Office Action, all of those claims stand rejected based on prior art rejections in which the Shah reference forms the primary prior art reference. However, as explained in response to the previous Office Action, Shah is not prior art because the inventors conceived and reduced to practice the present invention prior to the effective prior art date of the Shah reference, namely, November 6, 2000.

The claims remain rejected in this latest Office Action because the Office has deemed the declaration under 37 C.F.R. §1.131 to be deficient. Particularly, the Office asserted that the declaration merely states that Exhibit A describes the invention claimed in at least the rejected claims with no further explanation. As such, the Office deems the declaration insufficient per MPEP §715.07(I, last paragraph). In addition, it appears that the Office is asserting a second deficiency in the declaration. Particularly, in Section 1(a) of the Office Action, the Office asserted:

> The Applicants' 37 C.F.R. §1.131 Affidavit is signed by only three inventors, except Shriver, Elizabeth. Even so, Shriver was deceased, however, since there is nothing of record in the application regarding the fourth inventor (i.e. shown to be deceased) MPEP §715.04 still applies. In the case of Jointed Applicant of Application is deceased, The remaining Applicant(s) is/are referred to MPEP §715.04(D) and 409.01(e) for further information.

The Office was kind enough to reproduce those two sections of the MPEP in the Office Action with added underlining to indicate the relevant portions. Particularly, MPEP §715.04(D) was reproduced as shown below:

> Further, where it is <u>shown</u> that a joint inventor is deceased, refuses to sign or is otherwise unavailable, the signatures of the remaining joint inventors are sufficient. However, the affidavit or declaration, even though signed by fewer than all of the inventors, <u>must show completion</u> of the invention by all of the joint inventors of the subject matter of the claim(s) under rejection. (Underlining added by the Examiner).

Based on the above, apparently, the Office requires further proof of the death of Elizabeth Shriver. Again, in order to remedy this, attached hereto is a copy of a declaration by co-inventor Eran Gabber attesting to knowledge of Ms. Shriver's death on a first-hand basis. This should cure any defect in association with proof of Ms. Shriver's death.

However, it is unclear if the Office considers there to be another deficiency with the declaration with respect to showing completion of the invention by all of the joint inventors of the subject matter of the claims. Applicant assumes that this portion of the MPEP was underlined simply to emphasize the first noted shortcoming above, i.e., the lack of a specific discussion showing that Exhibit A discloses all of the claim recitations. The cited case, <u>In Re Carlson</u>, merely stands for the proposition that the Affidavit must establish that the invention was completed prior to the effective date of the prior art reference. The problem in In Re Carlson was that the evidence showed that one of the joint inventors invented a clock case while another inventor invented a clock base. They each filed separate applications, which were rejected. They then filed a joint application, which was the subject of the In Re Carlson case. The Court found simply that there was no evidence that they had put the base and the case together prior to the effective prior art date of the reference. As stated in that case:

> This being a joint application, it follows that the invention or design must have been the joint product of the applicants, and not the individual product of one or the other. There is nothing in this affidavit which states, or even suggests, that Carlson and Stitt, working jointly, combined their conceptions of a design for a base and a clock case before the date of filing of their joint application. All they did, according to this record, was the production of a design for a base by Stitt, and the design for a case by Carlson. In attempting to swear back of the reference, under Rule 75, Carlson presents certain exhibits which he contends show the unitary design. However, when these physical exhibits are placed together, they do not show the unitary design which is disclosed in their joint application. It seems quite obvious, therefore, that they have failed to comply with the conditions of the rule which requires the proof of the completion of the invention before the date of the referenced publication.

Accordingly, if the Office is suggesting that there is some requirement under Rule 131 to establish who the inventors were, this is clearly erroneous. Certainly, there is no such requirement when all of the inventors sign, such that it is unlikely that there is any such requirement when less than all of the inventors sign. In Re Carlson does not stand for this proposition.

Thus, with respect to a detailed showing that Exhibit A attached to the previous amendment discloses the invention as claimed, please see the following chart listing each element of each of the claims in question individually along with the supporting disclosure in Exhibit A.

CLAIM RECITATION	CORRESPONDING DISCLOSURE EXHIBIT IN A
Claim 1 A file system for a client computer system	In this paper, we characterize the web proxy workload, describe the design of Hummingbird, a light-weight file system for web proxies (Abstract – page 1, lines 5-6)
which comprises main memory	Hummingbird stores two types of main objects in main memory; files and clusters (page 6, line 11)
and at least one secondary storage device	See "Disk" in Figure 2 and entire paper
where said file system is programmed to receive and service file requests	Entire paper
to control accesses (including reads and writes) to a main memory	Hummingbird uses locality hints generated by the proxy to pack files into large, fixed-sized extents called *clusters* (page 4, lines 24-25)
to group files together in clusters	Hummingbird uses locality hints generated by the proxy to pack files into large, fixed-sized extents called *clusters*. These are the unit of disk access. Therefore, reads and writes are large, amortizing disk positioning times and interrupt processing. (page 4, lines 24-27)
and to store and retrieve clusters from said at least one secondary storage device	Hummingbird stores two types of main objects in main memory: *files and clusters* (page 6, line 11) See also al of section 4.3 of paper
and where said file system comprises file system clustering logic which assists in said grouping of files together	When the file system is building a cluster it determines which files to add using an LRU ordering according to the

in clusters	last time the file had been finished being read (i.e., when the done_read_file() call was received). If the least-recently-used file has a list of collocated files, then these files are added to the cluster if they are in main memory. If a file is on the collocation list, and already has been added to a cluster, it can still be added to the current cluster if the file is in memory. (Page 6, lines 21-26).
By grouping together files likely to be likely to be requested from said file system in close temporal proximity	When the file system is building a cluster it determines which files to add using an LRU ordering according to the last time the file had been finished being read (i.e., when the done_read_file() call was received). If the least-recently-used file has a list of collocated files, then these files are added to the cluster if they are in main memory. If a file is on the collocation list, and already has been added to a cluster, it can still be added to the current cluster if the file is in memory. (Page 6, lines 21-26).
Claim 2 The file system of claim 1, further comprising: a library of functions provided to applications by said file system	We have built a simple, lightweight file system named Hummingbird that runs on top of a raw disk partition. (Page 1, lines 27-28)
a function which takes as arguments at least two file names, which, when called, indicates to said file system clustering logic that said at least two file names provided as arguments should be stored together in one cluster if possible.	To support locality among files, the application sends collocate_files (fnameA, fnameB) calls. The file system saves the mapping until the files needs to be written to disk. The assignment of files to clusters occurs as late as possible; files are grouped together into clusters when space is needed in main memory. At this point, the file system attempts to write fnameA and fnameB in the same cluster. (Page 6, lines 17-20)

Claim 3 The file system of claim 1, where said file system clustering logic examines historical calls to files in order to determine which files are likely to be requested from said file system in tandem or close temporal proximity.	When the file system is building a cluster it determines which files to add using an LRU ordering according to the last time the file had been finished being read (i.e., when the done_read_file() call was received). If the least-recently-used file has a list of collocated files, then these files are added to the cluster if they are in main memory. If a file is on the collocation list, and already has been added to a cluster, it can still be added to the current cluster if the file is in memory. (Page 6, lines 21-26).
Claim 4 The file system of claim 3, further, comprising, as part of a library of functions provided to applications by said file system, a collocation function which takes as arguments at least two file names, which, when called, indicates to said file system clustering logic that the files named by the file names provided as arguments should be stored together in one cluster if possible.	To support locality among files, the application sends collocate_files (fnameA, fnameB) calls. The file system saves the mapping until the files needs to be written to disk. The assignment of files to clusters occurs as late as possible; files are grouped together into clusters when space is needed in main memory. At this point, the file system attempts to write fnameA and fnameB in the same cluster. (Page 6, lines 17-20).
Claim 5 A caching proxy comprising a computer system utilizing the file system of claim 4,	Caching web proxies are computer systems dedicated to caching and delivering web content (Page 1, line 10)
And programmed to receive and server requests for data from a large distributed-data network	Caching web proxies are computer systems dedicated to caching and delivering web content (Page 1, line 10; N.B.: The world wide web is a large distributed-data network)

Claim 6 The caching proxy system of claim 5, where said computer system is programmed to utilize said collocation function to provide an indication to said file system that the files named by the file names provided as arguments should be stored together in one cluster if possible.	To support locality among files, the application sends collocate_files (fnameA, fnameB) calls. The file system saves the mapping until the files needs to be written to disk. The assignment of files to clusters occurs as late as possible; files are grouped together into clusters when space is needed in main memory. At this point, the file system attempts to write fnameA and fnameB in the same cluster. When the file system is building a cluster it determines which files to add using an LRU ordering according to the last time the file had been finished being read (i.e., when the done_read_file() call was received). If the least-recently-used file has a list of collocated files, then these files are added to the cluster if they are in main memory. If a file is on the collocation list, and already has been added to a cluster, it can still be added to the current cluster if the file is in memory. (Page 6, lines 17-26).
Claim 7 The caching proxy system of claim 5, where said large distributed-data network is the World Wide Web.	Caching web proxies are computer systems dedicated to caching and delivering web content (Page 1, line 10)
Claim 20 The file system of claim 1, further comprising a library of functions provided to applications by said file system	We have built a simple, lightweight file system library named Hummingbird that runs on top of a raw disk partition. (Page 1, lines 27-28)
comprising a write function which, when called, writes a given file directly	int write_nomen_file(char*fname, void* buf, siz_t sz)'

to said at least one secondary storage device	This function bypasses the main memory cache and writes a file directly to disk. This file is flagged so that when it is faulted in, it does not compete with other documents for cache space and is immediately released after the proxy issues the done_read_file(). (Page 6, lines 1-4).
<u>Claim 21</u> The file system of claim 20, where said write function, when called, removes the given file from memory	 int write_nomen_file(char*fname, void* buf, siz_t sz)' This function bypasses the main memory cache and writes a file directly to disk. This file is flagged so that when it is faulted in, it does not compete with other documents for cache space and is immediately released after the proxy issues the done_read_file(). (Page 6, lines 1-4)
<u>Claim 22</u> The file system of claim 20, where said writing of a given file directly to said at least one secondary storage device is delayed until more space is needed in said main memory.	 To support locality among files, the application sends collocate_files (fnameA, fnameB) calls. The file system saves the mapping until the files needs to be written to disk. The assignment of files to clusters occurs as late as possible; files are grouped together into clusters when space is needed in main memory. At this point, the file system attempts to write fnameA and fnameB in the same cluster. (Page 6, lines 17-20).
<u>Claim 27</u> The file system of claim 1, where said file system further comprises a daemon which groups files together in clusters and stores clusters to said at least one secondary storage device.	 Pack files daemon. If the amount of main memory used to stored files is at or over a tunable threshold, the pack_files_daemon() uses the file LRU list to create a cluster of files and write

	the cluster to disk. (Page 7, lines 33-34)
Claim 28 The file system of claim 27, where the operations of said daemon occur when more space is needed in said main memory.	Pack files daemon. If the amount of main memory used to stored files is at or over a tunable threshold, the pack_files_daemon() uses the file LRU list to create a cluster of files and write the cluster to disk. (Page 7, lines 33-34)
Claim 29 The file system of claim 27, where the operations of said daemon occur when said file system is idle.	These daemons run when the file system is idle in a round-robin fashion, and can be synchronously called by a function if needed. (Page 7, lines 30-31)
Claim 30 The file system of claim 27, where the operations of said daemon occur when more space is needed in said main memory or when said file system is idle.	These daemons run when the file system is idle in a round-robin fashion, and can be synchronously called by a function if needed. Pack files daemon. If the amount of main memory used to stored files is at or over a tunable threshold, the pack_files_daemon() uses the file LRU list to create a cluster of files and write the cluster to disk. (Page 7, lines 30-34)

In view of the foregoing amendments and remarks, this application is now in condition for allowance. Applicant respectfully requests the Examiner to issue a Notice of Allowance at the earliest possible date. The Examiner is invited to contact Applicant's undersigned counsel by telephone call in order to further the prosecution of this case in any way.

Respectfully submitted,

Dated: 4.29.05

[illegible]

Theodore Naccarella
Registration No. 33,023
Synnestvedt & Lechner
2600 Aramark Tower
1101 Market Street
Philadelphia, PA 19107
Telephone: (215) 923-4466
Facsimile: (215) 923-2189

M:\TNACCARELLA\CLIENTS\LUCENT\25748\PTO\RESPONSE TO 1-31-05 OFFICE ACTION (DAS).DOC